-- Создание базы данных

CREATE DATABASE Шахматный\_клуб;

GO

-- Использование базы данных

USE Шахматный\_клуб;

GO

-- Создание таблицы "Клубы"

CREATE TABLE Клубы (

id\_клуба INT PRIMARY KEY IDENTITY(1, 1),

Название VARCHAR(100) NOT NULL,

Адрес VARCHAR(255) NOT NULL,

Телефон VARCHAR(20) NOT NULL

);

GO

-- Создание таблицы "Турниры"

CREATE TABLE Турниры (

id\_турнира INT PRIMARY KEY IDENTITY(1, 1),

Название VARCHAR(100) NOT NULL,

Место\_проведения VARCHAR(100) NOT NULL,

Дата\_начала DATE NOT NULL,

Дата\_окончания DATE NOT NULL

);

GO

-- Создание таблицы "Игроки"

CREATE TABLE Игроки (

id\_игрока INT PRIMARY KEY IDENTITY(1, 1),

Имя VARCHAR(100) NOT NULL,

Фамилия VARCHAR(100) NOT NULL,

Рейтинг INT NOT NULL,

id\_клуба INT,

FOREIGN KEY (id\_клуба) REFERENCES Клубы(id\_клуба)

);

GO

-- Создание таблицы "Участники\_турниры"

CREATE TABLE Участники\_турниры (

id\_турнира INT,

id\_игрока INT,

FOREIGN KEY (id\_турнира) REFERENCES Турниры(id\_турнира),

FOREIGN KEY (id\_игрока) REFERENCES Игроки(id\_игрока)

);

GO

-- Создание таблицы "Результаты"

CREATE TABLE Результаты (

id\_турнира INT,

id\_игрока INT,

Место INT NOT NULL,

Очки FLOAT,

Партии\_сыграно INT,

PRIMARY KEY (id\_турнира, id\_игрока),

FOREIGN KEY (id\_турнира) REFERENCES Турниры(id\_турнира),

FOREIGN KEY (id\_игрока) REFERENCES Игроки(id\_игрока)

);

GO

-- Добавление данных в таблицу "Клубы"

INSERT INTO Клубы (Название, Адрес, Телефон)

VALUES

('Клуб 1', 'Адрес 1', 'Телефон 1'),

('Клуб 2', 'Адрес 2', 'Телефон 2'),

('Клуб 3', 'Адрес 3', 'Телефон 3'),

('Клуб 4', 'Адрес 4', 'Телефон 4'),

('Клуб 5', 'Адрес 5', 'Телефон 5');

-- Добавление данных в таблицу "Турниры"

INSERT INTO Турниры (Название, Место\_проведения, Дата\_начала, Дата\_окончания)

VALUES

('Турнир 1', 'Место 1', '2023-01-01', '2023-01-05'),

('Турнир 2', 'Место 2', '2023-02-01', '2023-02-05'),

('Турнир 3', 'Место 3', '2023-03-01', '2023-03-05'),

('Турнир 4', 'Место 4', '2023-04-01', '2023-04-05'),

('Турнир 5', 'Место 5', '2023-05-01', '2023-05-05');

-- Добавление данных в таблицу "Игроки"

INSERT INTO Игроки (Имя, Фамилия, Рейтинг, id\_клуба)

VALUES

('Игрок 1', 'Фамилия 1', 1500, 1),

('Игрок 2', 'Фамилия 2', 1600, 2),

('Игрок 3', 'Фамилия 3', 1700, 3),

('Игрок 4', 'Фамилия 4', 1800, 4),

('Игрок 5', 'Фамилия 5', 1900, 5);

-- Добавление данных в таблицу "Участники\_турниры"

INSERT INTO Участники\_турниры (id\_турнира, id\_игрока)

VALUES

(1, 1),

(1, 2),

(2, 3),

(2, 4),

(3, 5);

-- Добавление данных в таблицу "Результаты"

INSERT INTO Результаты (id\_турнира, id\_игрока, Место, Очки, Партии\_сыграно)

VALUES

(1, 1, 1, 1.0, 10),

(1, 2, 2, 0.5, 10),

(2, 3, 1, 1.0, 10),

(2, 4, 2, 0.5, 10),

(3, 5, 1, 1.0, 10);

CREATE VIEW Список\_игроков\_и\_клубов

AS

SELECT Игроки.Имя, Игроки.Фамилия, Клубы.Название AS Клуб

FROM Игроки

JOIN Клубы ON Игроки.id\_клуба = Клубы.id\_клуба;

CREATE VIEW Турниры\_и\_количество\_участников

AS

SELECT Турниры.Название, COUNT(Участники\_турниры.id\_турнира) AS Количество\_участников

FROM Турниры

LEFT JOIN Участники\_турниры ON Турниры.id\_турнира = Участники\_турниры.id\_турнира

GROUP BY Турниры.id\_турнира, Турниры.Название;

CREATE VIEW Результаты\_игроков\_в\_турнирах

AS

SELECT Игроки.Имя, Игроки.Фамилия, Турниры.Название AS Турнир, Результаты.Место, Результаты.Очки

FROM Игроки

JOIN Результаты ON Игроки.id\_игрока = Результаты.id\_игрока

JOIN Турниры ON Результаты.id\_турнира = Турниры.id\_турнира;

CREATE PROCEDURE Добавить\_нового\_игрока

@Имя VARCHAR(100),

@Фамилия VARCHAR(100),

@Рейтинг INT,

@id\_клуба INT

AS

BEGIN

INSERT INTO Игроки (Имя, Фамилия, Рейтинг, id\_клуба)

VALUES (@Имя, @Фамилия, @Рейтинг, @id\_клуба);

END;

CREATE PROCEDURE Обновить\_рейтинг\_игрока

@id\_игрока INT,

@Новый\_рейтинг INT

AS

BEGIN

UPDATE Игроки

SET Рейтинг = @Новый\_рейтинг

WHERE id\_игрока = @id\_игрока;

END;

CREATE PROCEDURE Получить\_список\_турниров\_и\_участников

AS

BEGIN

SELECT Турниры.Название, COUNT(Участники\_турниры.id\_турнира) AS Количество\_участников

FROM Турниры

LEFT JOIN Участники\_турниры ON Турниры.id\_турнира = Участники\_турниры.id\_турнира

GROUP BY Турниры.id\_турнира, Турниры.Название;

END;

USE Шахматный\_клуб;

GO

CREATE TRIGGER Ограничение\_Количества\_Участников

ON Участники\_турниры

AFTER INSERT

AS

BEGIN

DECLARE @id\_турнира INT;

SELECT @id\_турнира = id\_турнира FROM INSERTED;

DECLARE @количество\_участников INT;

SELECT @количество\_участников = COUNT(\*) FROM Участники\_турниры WHERE id\_турнира = @id\_турнира;

IF @количество\_участников > 100

BEGIN

RAISERROR('Достигнуто максимальное количество участников для данного турнира.', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END;

END;

GO

CREATE TRIGGER Ограничение\_на\_удаление\_клуба

ON Клубы

INSTEAD OF DELETE

AS

BEGIN

DECLARE @id\_клуба INT;

SELECT @id\_клуба = id\_клуба FROM DELETED;

IF EXISTS (SELECT \* FROM Игроки WHERE id\_клуба = @id\_клуба)

BEGIN

RAISERROR ('Невозможно удалить клуб, так как в нем зарегистрированы игроки.', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END;

DELETE FROM Клубы WHERE id\_клуба = @id\_клуба;

END;

-- Создание триггера "Обновление\_рейтинга\_игрока"

CREATE TRIGGER Обновление\_рейтинга\_игрока

ON Результаты

AFTER INSERT, UPDATE

AS

BEGIN

-- Обновление рейтинга игрока при добавлении или изменении результата

UPDATE Игроки

SET Рейтинг = (

SELECT AVG(Очки) FROM Результаты WHERE id\_игрока = Игроки.id\_игрока

)

WHERE id\_игрока IN (SELECT id\_игрока FROM inserted);

END;

CREATE FUNCTION ПолучитьУчастниковТурнира(@id\_турнира INT)

RETURNS TABLE

AS

RETURN (

SELECT Игроки.\*

FROM Игроки

INNER JOIN Участники\_турниры ON Игроки.id\_игрока = Участники\_турниры.id\_игрока

WHERE Участники\_турниры.id\_турнира = @id\_турнира

);

CREATE FUNCTION ПолучитьРезультатыТурнира(@id\_турнира INT)

RETURNS TABLE

AS

RETURN (

SELECT Результаты.\*

FROM Результаты

WHERE Результаты.id\_турнира = @id\_турнира

);

CREATE FUNCTION ПолучитьКлубИгрока(@id\_игрока INT)

RETURNS TABLE

AS

RETURN (

SELECT Клубы.\*

FROM Клубы

INNER JOIN Игроки ON Клубы.id\_клуба = Игроки.id\_клуба

WHERE Игроки.id\_игрока = @id\_игрока

);

1. **Перечислите основные принципы построения концептуальной, логической и физической модели данных.**

Основные принципы построения концептуальной, логической и физической моделей данных включают:

Концептуальная модель данных:

• Абстракция: Она должна описывать сущности, их атрибуты и связи на высоком уровне, отражая предметную область без учета деталей хранения данных.

• Унификация: Концептуальная модель должна объединять данные из различных источников и представлять общую семантику и структуру данных.

• Простота и понятность: Модель должна быть простой и легко понятной для заинтересованных сторон, включая аналитиков, разработчиков и пользователей.

Логическая модель данных:

• Структурированность: Она должна быть организована таким образом, чтобы обеспечить эффективное хранение и доступ к данным.

• Нормализация: Нормализация помогает устранить аномалии и повысить эффективность работы с данными.

• Согласованность: Логическая модель должна быть согласованной с концептуальной моделью и отражать ее семантику и структуру.

Физическая модель данных:

• Оптимизация: Она должна быть оптимизирована для обеспечения высокой производительности и эффективности работы с данными.

• Индексирование: Индексы помогают сократить время выполнения запросов и улучшить общую производительность системы.

• Физические ограничения: Физическая модель определяет ограничения целостности данных, такие как уникальность, ссылочная целостность и другие правила, которые должны соблюдаться при работе с данными на физическом уровне.